IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Patent Application of
TARBELL et al.
Serial No. 10/531,260
Filed: April 13, 2005
For: METHOD, SYSTEM AND SOFTWARE FOR JOURNALING SYSTEM OBJECTS

Atty. Ref.: PTB-4942-5
TC/A.U.: 2164
Examiner: Yuk Ting Choi

* *

March 1, 2010

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

Appellant hereby **appeals** to the Board of Patent Appeals and Interferences from the last decision of the Examiner. Petition for a two-month extension of time is hereby made.

03/02/2010 HVUONG1 00000054 10531260
01 FC:2402 270.00 OP
03/02/2010 HVUONG1 00000054 10531260
02 FC:2252 245.00 OP

TABLE OF CONTENTS

(I) REAL PARTY IN INTEREST

The real party in interest is Maximum Availability Limited, a corporation of the country of New Zealand, by way of assignment recorded at Reel 017273, Frame 0601.

(II) RELATED APPEALS AND INTERFERENCES

The appellant, the undersigned, and the assignee are not aware of any related appeals, interferences, or judicial proceedings (past or present), which will directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

(III) STATUS OF CLAIMS

Claims 46-91 are pending and have been rejected. Claims 1-45 and 92-93 previously were cancelled. No claims have been substantively allowed. The rejections of claims 46-91 are being appealed.

(IV) STATUS OF AMENDMENTS

No amendments have been filed since the date of the last Rejection (i.e., since the Office Action dated July 27, 2009).

(V) SUMMARY OF CLAIMED SUBJECT MATTER

The following summary is for illustrative and explanatory purposes only. The references to the specification and drawings are not an admission that the claimed inventions are limited to any, or all, of the disclosed embodiments. The references to one, or plural, embodiment(s) is not an admission that the claimed features are only described in relation to that one, or plural, embodiment(s).

46. A computer implemented method of journaling in a database (e.g., 20 in Fig. 1) journal changes to system objects (e.g., 16 in Fig. 1) in an operating system with a processor (e.g., see Figs. 1 and 2; p. 6, lines 3-8), the method including:

i) executing a dummy function (e.g., 4 in Fig. 1) in place of a system function (e.g., 5 in Fig. 1) when the system function is called (p. 3, lines 19-21; p. 6, lines 12-25);

ii) executing the system function (e.g., 5 in Fig. 1) under operation of the dummy function (e.g., p. 3, line 22; p. 6, lines 26-33); and generating copies of system objects (e.g., 19 in Fig. 1), changed by the execution of the system function (e.g., 5 in Fig. 1), for journaling (e.g., p. 3, lines 23-25; p. 8, lines 20-26).

47. A method as claimed in claim 46 wherein the dummy function (e.g., 4 in Fig. 1) is executed in place of the system function (e.g., 5 in Fig. 1) by assigning a duplicate calling name to the dummy function and arranging the processor to pre-empt the execution of the system function when it is called (e.g., p. 3, lines 27-28; p. 6, lines 12-13).

48. A method as claimed in claim 47 wherein the dummy function (e.g., 4 in Fig. 1) includes an exit point (e.g., 9 in Fig. 2), and an exit program is registered for the exit point (e.g., p. 6, lines 26-33).

49. A method as claimed in claim 48 wherein during operation of the dummy function (e.g., 4 in Fig. 1) the exit program is executed upon reaching the exit point (e.g., 13 in Fig. 2; p. 7, lines 1-4).

53. A method as claimed in claim 49 wherein the execution of the system function (e.g., 5 in Fig. 1) is handled by the dummy function (e.g., 4 in Fig. 1; p. 3, line 22; p. 6, lines 26-33).

68. A system (e.g., see Figs. 1 and 2; p. 5, lines 10-15; p. 6, lines 3-8) for journaling in a database (e.g., 20 in Fig. 1) journal changes to system objects (e.g., 16 in Fig. 1) including:

i) a processor adapted to execute a dummy function (e.g., 4 in Fig. 1) in place of a system function (e.g., 5 in Fig. 1) when the system function is called (p. 3, lines 19-21; p. 6, lines 12-25), wherein the dummy function (e.g., 4 in Fig. 1) executes the system function (e.g., 5 in Fig. 1) and generates copies of system objects (e.g., 19 in Fig. 1) resulting from the execution of the system function execution for journaling (e.g., p. 3, lines 19-25; p. 6, lines 26-33); and

ii) memory for use by the processor during execution (e.g., p. 5, line 15).

69. A system as claimed in claim 68 wherein the dummy function (e.g., 4 in Fig. 1) is executed in place of the system function (e.g., 5 in Fig. 1) by assigning a duplicate calling name to the dummy function and arranging the processor to pre-empt the execution of the system function when it is called (e.g., p. 3, lines 27-28; p. 6, lines 12-13).

70. A system as claimed in claim 69 wherein the dummy function (e.g., 4 in Fig. 1) includes an exit point (e.g., 9 in Fig. 2), and an exit program is registered for the exit point (e.g., p. 6, lines 26-33).

71. A system as claimed in claim 70 wherein during operation of the dummy function (e.g., 4 in Fig. 1) the exit program is executed upon reaching the exit point (e.g., 13 in Fig. 2; p. 7, lines 1-4).

75. A system as claimed in claim 71 wherein the execution of the system function (e.g., 5 in Fig. 1) is handled by the dummy function (e.g., 4 in Fig. 1; p. 3, line 22; p. 6, lines 26-33).

(VI) GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

First, claims 46-56, 62-66, 68-78, and 84-91 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Bills (U.S. Publication No. 2002/0152195) in view of Tanaka (U.S. Patent No. 6,665,735).

Second, claims 57-59 and 79-81 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Bills and Tanaka, and further in view of Owen (U.S. Publication No. 2003/0217031).

Third, claims 60-61 and 82-83 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Bills and Tanaka, and further in view of Suzuki (U.S. Patent No. 6,829,768).

(VII) ARGUMENT

A. Claims 46-56, 62-66, 68-78, and 84-91 Are Not Unpatentable Over Bills and Tanaka.

Claims 46-56, 62-66, 68-78, and 84-91 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Bills (U.S. Publication No. 2002/0152195) in view of Tanaka (U.S. Patent No. 6,665,735). This rejection is erroneous and should be reversed for at least the following reasons.

The instant application includes two independent claims (i.e., claims 46 and 68). Claim 46 is directed to a computer implemented method of journaling in a database journal changes to system objects in an operating system with a processor, the method including i) executing a dummy function in place of a system function when the system function is called, ii) executing the system function under operation of the dummy function, and generating copies of system objects, changed by the execution of the system function, for journaling.

Similarly, claim 68 is directed to a system for journaling in a database journal changes to system objects including i) a processor adapted to execute a dummy function in place of a system function when the system function is called, wherein the dummy function executes the system function and generates copies of system objects resulting from the execution of the system function execution for journaling, and ii) memory for use by the processor during execution.

Bills teaches a method of recording data changes to objects. The method includes the steps of determining if there is an indication within the object that any changes to that

object should not be stored in the object. If there is such a determination, then the changes are stored only in a journal entry in a journal. Therefore, Bills is not concerned with storing objects after they have changed. Instead, Bills is solely concerned with avoiding storing those changed objects. This is clearly different from claims 46 and 68, wherein copies of system objects are changed by the execution of a system function, and those changed system objects are generated for journaling. Bills does not disclose generating copies of system objects, changed by the execution of the system function, for journaling. Bills only discloses a journal for storing changes made to the objects, not generating copies of changed system objects.

The introduction of the instant application emphasizes that the differences between selectively storing changes to a journal entry (as in Bills) and generating copies of system objects, changed by the execution of the system function, for journaling (as in claims 46 and 68) have been contemplated. For example, in discussing techniques similar to those disclosed in Bills, the specification notes that:

> "The creation, deletion and changing of system objects may be recorded in the Audit Journal for the primary purpose of providing an audit of activity related to these objects. When viewed with the intention of providing replication of these objects to a remote or local (copy) the Audit Journal has several significant drawbacks. . . ."

The specification then goes on to list four significant drawbacks of these types of systems. See pages 1-2 of the specification in this regard.

The invention defined by the claims provides advantages over the journaling prior art in that journaling changes are able to be made without being restrained to the journaling of particular objects as defined by the current database version, whereas the

journaling prior art is focused on journaling system objects only for a portion of defined objects as defined by the current database version. Indeed, from this prior art description, it is clear that claims 46 and 68 provide useful solutions in the art of database journaling that clearly was not available prior to its inception, e.g., that of providing a general manner of creating journaling functionality without being restrained to particular defined objects.

The Office Action argues that paragraphs 5 and 14 of Bills correspond to the above-identified features of the independent claims. These paragraphs describe a journal file system (JFS). According to Bills, a JFS can 1) record changes to objects, 2) enable single system recovery, and 3) provide for recovery of a saved object to a known state. Paragraphs 9-11 describe these "areas of support," which merely relate to techniques for leveraging the record of changes that are made to journaled objects. Indeed, paragraphs 9-11 make clear that it is the changes to the objects that are recorded and used by Bills -- not the changed objects themselves.

The Office Action also cites to paragraph 58 of Bills for the generating copies of system objects for journaling. This paragraph contemplates creating new objects and modifying existing objects, which ultimately could lead to a journaling step (see step 518 in Fig. 5 of Bills), depending on the circumstances. Bills makes clear that such a journaling step creates a journaling record 61, which is a part of its journal log 60, as shown in Fig. 2 thereof. This journal log 60 is a part of Bills' JFS 56. Bills does not, however, teach or suggest generating copies of system objects for journaling as a result of creating a new object or modifying an existing object in this journal log 60 -- or

anywhere else in its rather conventional JFS 56. In other words, the JFS 56 of Bills does not store objects after they have changed; it merely keeps a record of the changes that were made and stores an indication as to "whether a journal record 61 should be journaled upon the occurrence a predefined event affecting an object" (paragraph 45).

The Office Action therefore errs, as Bills does not teach or suggest generating copies of system objects, changed by the execution of the system function, for journaling. Citation to the Tanaka does not cure this fundamental deficiency of Bills. Reversal of the § 103 rejections therefore is respectfully requested for at least this reason.

Of course, in addition to the above, the Office Action admits that Bills does not teach or suggest the steps within a method of journaling changes to system objects in a database journal of executing a dummy function in place of a system function when the system function is called, where the system function is then executed under operation of the dummy function. The Office Action thus cites to Tanaka to make up for this admitted deficiency of Bills. However, Applicant respectfully submits that Tanaka is not analogous art. Tanaka lies within a completely different technical field that is wholly unrelated to a method of journaling changes to system objects in database journals to which claims 46 and 68 are directed.

Indeed, Tanaka teaches a solution in the art of linking and compiling programming objects, which is wholly unrelated to database journaling. A person skilled in the database journaling art would not reasonably look to the art of linking and compiling to find a solution to the problem of synchronizing system object changes with associated database changes using an Audit Journal.

That is, Tanaka is concerned with making changes to libraries and modules at a system level for the purposes of programming, whereas certain embodiments of the claimed invention are directed to journaling system objects in database journaling. Furthermore, certain embodiments of the claimed invention are directed to the execution of a dummy function as a replacement to a system function as a whole, which would be understood by the skilled person to be at a user level, rather than a system level (i.e., a kernel level) as described in Tanaka. Therefore, the inventions of claims 46 and 68 are implemented at higher functional levels as compared to that of Tanaka. These fundamental differences make the combination of Tanaka with a database journaling system wholly incompatible.

Tanaka clearly is different in structure and function. It is perhaps not that surprising, then, that the USPTO itself has placed the instant application, Bills, and Tanaka in wholly different classes and subclasses. In a nutshell, because Tanaka is related to problems wholly unrelated to database journaling, it is clear that (1) Tanaka is not "reasonably pertinent" to Applicant's field of endeavor, (2) would not have "logically . . . commended itself to an inventor's attention in considering his or her invention as a whole," and therefore (3) cannot be considered analogous art.

This error is borne out in the alleged motivation for making the combination. One of ordinary skill in the art at the time of the invention would not have modified Bills in view of Tanaka in the manner alleged in the Office Action "to expand a programming function without altering the original programming function of the system." Indeed, this rationale makes no sense in the context of prior art that it is not at all related to linking

and compiling programming objects, or in the context of the instant disclosure, which is directed to solving the problems of synchronizing system object changes with associated database changes using an Audit Journal. This sort of ex post reconstruction of Applicant's claimed invention is impermissible. Thus, the § 103 rejections are improper for these additional reasons.

In view of the above, Applicant respectfully submits that the outstanding rejections are erroneous and should be reversed.

Claims 47 and 69

Claims 47 and 69 (and their respective dependents) should be allowable over the alleged Bills/Tanaka combination for further reasons. In particular, claims 47 and 69 both specify that "the dummy function is executed in place of the system function by assigning a duplicate calling name to the dummy function and arranging the processor to pre-empt the execution of the system function when it is called." Tanaka cannot possibly teach or suggest this subject matter. In particular, Tanaka cannot pre-empt the execution of the system function when a dummy function is called because Tanaka is concerned with design-time functionality, not run-time functionality. Thus, the rejection of claims 47 and 69 (and their respective dependents) is flawed for this further reason.

Claims 53 and 75

Claims 53 and 75 (and their respective dependents) also should be allowable over the alleged Bills/Tanaka combination for further reasons. In particular, claims 53 and 75

both specify that "the execution of the system function is handled by the dummy function." Similar to the above, Tanaka cannot possibly teach or suggest this subject matter, because its dummy functions are provided at design-time, not run-time. Thus, the rejection of claims 53 and 75 (and their respective dependents) is flawed for this further reason.

B. Claims 57-59 and 79-81 Are Not Unpatentable Over Bills, Tanaka, and Owen.

Claims 57-59 and 79-81 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Bills and Tanaka, and further in view of Owen (U.S. Publication No. 2003/0217031). Fundamental deficiencies with Bills, Tanaka, and the alleged Bills/Tanaka combination have been described in detail above. The further introduction of Owen does not make up for these fundamental deficiencies. Thus, this rejection is erroneous and should be reversed.

A. Claims 60-61 and 82-83 Are Not Unpatentable Over Bills, Tanaka, and Suzuki.

Claims 60-61 and 82-83 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Bills and Tanaka, and further in view of Suzuki (U.S. Patent No. 6,829,768). Fundamental deficiencies with Bills, Tanaka, and the alleged Bills/Tanaka combination have been described in detail above. The further introduction of Suzuki does not make up for these fundamental deficiencies. Thus, this rejection is erroneous and should be reversed.

CONCLUSION

In conclusion it is believed that the application is in clear condition for allowance; therefore, early reversal of the Rejection and passage of the subject application to issue are earnestly solicited.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: _____

Paul T. Bowen
Reg. No. 38,009

PTB:jr
901 North Glebe Road, 11th Floor
Arlington, VA 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100

(VIII) CLAIMS APPENDIX

46. A computer implemented method of journaling in a database journal changes to system objects in an operating system with a processor, the method including:

i) executing a dummy function in place of a system function when the system function is called;

ii) executing the system function under operation of the dummy function; and generating copies of system objects, changed by the execution of the system function, for journaling.

47. A method as claimed in claim 46 wherein the dummy function is executed in place of the system function by assigning a duplicate calling name to the dummy function and arranging the processor to pre-empt the execution of the system function when it is called.

48. A method as claimed in claim 47 wherein the dummy function includes an exit point, and an exit program is registered for the exit point.

49. A method as claimed in claim 48 wherein during operation of the dummy function the exit program is executed upon reaching the exit point.

50. A method as claimed in claim 49 wherein the execution of the system function is handled by the exit program.

51. A method as claimed in claim 50 wherein the system objects changed by execution of the system function are captured by the exit program.

52. A method as claimed in claim 51 wherein the exit program generates copies of the system objects captured by the exit program.

53. A method as claimed in claim 49 wherein the execution of the system function is handled by the dummy function.

54. A method as claimed in claim 53 wherein the system objects changed by execution of the system function are captured by the dummy function.

55. A method as claimed in claim 54 wherein the exit program generates copies of the system objects captured by the dummy function.

56. A method as claimed in claim 52 wherein the copies of the system objects are saved to disk.

57. A method as claimed in claim 52 wherein the copies of the system objects are streamed to a database system for journaling.

58. A method as claimed in claim 57 wherein the database system is incorporated with a replication system.

59. A method as claimed in claim 58 wherein the replication system replicates the copies of the system objects to one or more local or remote databases.

60. A method as claimed in claim 52 wherein messages or exceptions generated by the system function are captured into a queue.

61. A method as claimed in claim 60 wherein the system function is originally called by a process and the messages or exceptions are forwarded back to the process by the dummy function.

62. A method as claimed in claim 52 wherein the system objects are one or more of the set of program objects, configuration objects, queues, and space/memory mapped objects.

63. A method as claimed in claim 52 wherein the changed system objects are those system objects which have been created, changed or deleted.

64. A method as claimed in claim 52 wherein the system functions are OS/400 system functions.

65. The method as claimed in claim 46 further including:

i) executing the system function during which changes to system objects occur; and

ii) journaling changes to system objects during execution of the system function.

66. A method as claimed in claim 65 wherein changes to system objects are journaled by integrating journaling commands into the code of the system functions.

67. A method as claimed in claim 65 wherein changes to system objects are journaled by associating exit points with the system function and calling an exit program during execution of the system function.

68. A system for journaling in a database journal changes to system objects including:

i) a processor adapted to execute a dummy function in place of a system function when the system function is called, wherein the dummy function executes the system function and generates copies of system objects resulting from the execution of the system function execution for journaling; and

ii) memory for use by the processor during execution.

69. A system as claimed in claim 68 wherein the dummy function is executed in place of the system function by assigning a calling name to the dummy function that is a duplicate of the system function calling name to pre-empt the execution of the system function.

70. A system as claimed in claim 69 wherein the dummy function includes an exit point, and an exit program is registered for the exit point.

71. A system as claimed in claim 70 wherein during execution of the dummy function the exit program is executed upon reaching the exit point.

72. A system as claimed in claim 71 wherein the execution of the system function is handled by the exit program.

73. A system as claimed in claim 72 wherein the system objects changed by execution of the system function are captured by the exit program.

74. A system as claimed in claim 72 wherein the exit program generates copies of the system objects captured by the exit program.

75. A system as claimed in claim 71 wherein the execution of the system function is handled by the dummy function.

76. A system as claimed in claim 75 wherein the system objects changed by execution of the system function are captured by the dummy function.

77. A system as claimed in claim 76 wherein the exit program generates copies of the system objects captured by the dummy function.

78. A system as claimed in claim 74 wherein the copies of the system objects are saved to disk.

79. A system as claimed in claim 74 wherein the copies of the system objects are streamed to a database system for journaling.

80. A system as claimed in claim 79 wherein the database system is incorporated with a replication system.

81. A system as claimed in claim 80 wherein the replication system replicates the copies of the system objects to one or more local or remote databases.

82. A system as claimed in claim 74 wherein messages or exceptions generated by the system function are captured into a queue.

83. A system as claimed in claim 82 wherein the system function is originally called by a process and the messages or exceptions are forwarded back to the process by the dummy function.

84. A system as claimed in claim 74 wherein the system objects are one or more of the set of program objects, configuration objects, queues, and space/memory mapped objects.

85. A system as claimed in claim 74 wherein the changed system objects are those system objects which have been created, changed or deleted.

86. A system as claimed in claim 74 wherein the processor is an AS/400 processor.

87. A system as claimed in claim 86 wherein the processor is operating under the OS/400 operating system.

88. A computer system for effecting the method of claim 46.

89. A computer system for effecting the method of claim 65.

90. A computer readable storage medium tangibly storing software executable by a computer for executing the method of claim 46.

91. A computer readable storage medium tangibly storing software executable by a computer for executing the method of claim 65.

(X) RELATED PROCEEDINGS APPENDIX

None.